AFHRL-TP-88-8

DTIC FILE COPY (2)

AIR FORCE

HUMAN RESOURCES

AD-A197 971

DEVELOPMENT AND DEMONSTRATION OF MICROCOMPUTER INTELLIGENCE FOR TECHNICAL TRAINING (MITT)

William B. Johnson
Ruston M. Hunt
Phillip C. Duncan
Jeffrey E. Norton

Search Technology, Incorporated
Suite 200
4725 Peachtree Corners Circle
Norcross, Georgia 30092

TRAINING SYSTEMS DIVISION
Brooks Air Force Base, Texas 78235-5601

August 1988

Final Technical Paper for Period July 1987 - January 1988

Approved for public release; distribution is unlimited.

LABORATORY

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5601

88 8 08 145

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this paper, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This paper has been reviewed and is approved for publication.

CHARLES G. CAPPS, 1Lt, USAF
Contract Monitor

HENDRICK W. RUCK, Technical Advisor
Training Systems Division

GENE A. BERRY, Colonel, USAF
Chief, Training Systems Division

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified

1b. RESTRICTIVE MARKINGS:

2a. SECURITY CLASSIFICATION AUTHORITY:

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE:

3. DISTRIBUTION / AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S):

5. MONITORING ORGANIZATION REPORT NUMBER(S): AFHRL-TP-88-8

6a. NAME OF PERFORMING ORGANIZATION: Search Technology, Incorporated

6b. OFFICE SYMBOL (If applicable):

7a. NAME OF MONITORING ORGANIZATION: Training Systems Division

6c. ADDRESS (City, State, and ZIP Code):
Suite 200
4725 Peachtree Corners Circle
Norcross, Georgia 30092

7b. ADDRESS (City, State, and ZIP Code):
Air Force Human Resources Laboratory
Brooks Air Force Base, Texas 78235-5601

8a. NAME OF FUNDING / SPONSORING ORGANIZATION: Air Force Human Resources Laboratory

8b. OFFICE SYMBOL (If applicable): HQ AFHRL

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F41689-86-D-0052

8c. ADDRESS (City, State, and ZIP Code): Brooks Air Force Base, Texas 78235-5601

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO
62703F	7734	13	01

11. TITLE (Include Security Classification): Development and Demonstration of Microcomputer Intelligence for Technical Training (MITT)

12. PERSONAL AUTHOR(S): Johnson, W.B.; Hunt, R.M.; Duncan, P.C.; Norton, J.E.

13a. TYPE OF REPORT: Final

13b. TIME COVERED: FROM Jul 87 TO Jan 88

14. DATE OF REPORT (Year, Month, Day): August 1988

15. PAGE COUNT: 44

16. SUPPLEMENTARY NOTATION: Under subcontract to: Universal Energy Systems, Incorporated, 4401 Dayton-Xenia Road, Dayton, Ohio 45432.

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
12	09	
05	08	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number):
artificial intelligence, intelligent tutoring systems, computer-based instruction, training devices. (SDW) expert systems,

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The Microcomputer Intelligence for Technical Training research described here is founded upon a proven intelligent diagnostic training simulation to construct enhanced models of the student, instructor, and expert in a training environment. An off-the-shelf microcomputer was used to deliver an operational intelligent tutoring system (ITS) within 180 days of project start.

The demonstration ITS was developed in cooperation with Air Force and NASA personnel at the Lyndon B. Johnson Space Center in Houston, Texas. The space shuttle fuel cell was the technical domain for this first application. Initial trial use by astronauts, flight controllers, and technical training personnel indicates that the approach is technically feasible and has instructional value for space training applications.

Keywords:

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION:

22a. NAME OF RESPONSIBLE INDIVIDUAL: Nancy J. Allin, Chief, STINFO Office

22b. TELEPHONE (Include Area Code): (512) 536-3877

22c. OFFICE SYMBOL: AFHRL/TSR

DD Form 1473, JUN 86 — Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

AFHRL Technical Paper 88-8

August 1988

DEVELOPMENT AND DEMONSTRATION OF MICROCOMPUTER INTELLIGENCE FOR TECHNICAL TRAINING (MITT)

William B. Johnson
Ruston M. Hunt
Phillip C. Duncan
Jeffrey E. Norton

Search Technology, Incorporated
Suite 200
4725 Peachtree Corners Circle
Norcross, Georgia 30092

TRAINING SYSTEMS DIVISION
Brooks Air Force Base, Texas 78235-5601

Reviewed and submitted for publication by

Hugh L. Burns, Jr., Lt Col, USAF
Chief, Intelligent Systems Branch

This publication is primarily a working paper. It is published solely to document work performed.

SUMMARY

The Microcomputer Intelligence for Technical Training research described here is founded upon a proven intelligent diagnostic training simulation to construct enhanced models of the student, instructor and expert in a training environment. An off-the-shelf microcomputer was used to deliver an operational intelligent tutoring system (ITS) within 180 days of project start.

The demonstration ITS was developed in cooperation with Air Force and NASA personnel at the Lyndon B. Johnson Space Center in Houston, Texas. The space shuttle fuel cell was the technical domain for this first application. Initial trial use by astronauts, flight controllers and technical training personnel indicates that the approach is technically feasible and has instructional value for space training applications.

ACKNOWLEDGMENTS

The authors wish to acknowledge the Air Force Human Resources Laboratory, Lt Col Hugh Burns, Capt Kevin Kline, and Lt Charles Capps for their guidance and support. Their technical input and evaluation were critically important. We thank Ms. Barbara Pearson, Lt Dan Soderlund, and Mr. Ken Swiatkowski from Johnson Space Center Training Department. The development and numerous revisions of the multitude of computer graphics was accomplished by Ms. Connie McFarland of Search Technology, Inc.

TABLE OF CONTENTS

List of Figures

List of Tables

1.0 INTRODUCTION

Intelligent Tutoring System (ITS) research and development (R&D) has evolved as a combination of Computer-Based Instruction (CBI) and Artificial Intelligence (AI). The disciplines of Psychology, Education, and Computer Science have merged to produce concepts and systems that promise to affect training now and in the future. During this first 6-month phase of the research we accomplished the goal of using proven intelligent CBI to build and demonstrate a fully operational ITS on a microcomputer. This report describes the first Microcomputer Intelligence for Technical Training (MITT) product and discusses the development and demonstration within a National Aeronautics and Space Administration (NASA) technical domain related to the space shuttle fuel cell system.

2.0 BACKGROUND--INTELLIGENT TUTORING SYSTEMS

This section discusses ITSs as they currently exist. Also discussed is the computer-based diagnostic training research conducted by the authors since 1976. The purpose of reviewing this information is to summarize the lessons learned during that period and the state of the art at the outset of this project.

2.1 Existing Intelligent Tutoring Systems

Various state-of-the-art ITSs were discussed at two recent conferences sponsored by the United States Air Force Human Resources Laboratory (AFHRL) (Richardson & Polson, 1988) and the Army Research Institute for the Behavioral and Social Sciences (Psotka, Massey, & Mutter, 1988). A few of the most notable ITSs are reviewed below.

AI has influenced many fields including medicine (e.g., INTERNIST, CADUCEUS, MYCIN, and PUFF), chemistry (e.g., DENDRAL), biology (e.g., MOLGEN), geology (e.g., PROSPECTOR and DRILLING ADVISOR), communications diagnosis (e.g., ACE) and locomotive repair (e.g., DELTA/CATSI). These systems were designed as job decision aids and attempted to bring an expert to the job site, laboratory or clinic. These AI applications, however, were not designed for instruction and tutoring. The Air Force Integrated Maintenance Information System (IMIS), which is being developed, is another example of an AI project that emphasizes aiding rather than training.

Most expert systems are not designed to provide training. However, training professionals are quick to recognize the potential training value of programs with extensive, well-organized knowledge bases and algorithms for applying that knowledge to problems. These expert systems were not designed to explain their decision making process in a manner which is easily understood by students, particularly novice students.

Without the capability of explanation, the training value of these systems is low. It is obvious to training personnel that expert systems must have a number of features in order to provide a reasonable level of intelligent tutoring. Specifically, training systems must reflect an understanding of students, instructors/curriculum planners and experts who understand the content areas. These functions are commonly referred to as Student, Instructor and Expert modules. Therefore, in intelligent tutoring, development must capitalize on the research related to instruction, to expert systems and to work being done in the area of intelligent human-computer interfaces.

There are a number of ITSs in development today (Wenger, 1987). Table 1 depicts the most notable of the existing systems. For the most part, these systems were developed as laboratory tools and remain in the laboratory to test various hypotheses related to intelligent training systems. An exception is Anderson's LISP Tutor that has transcended initial laboratory evaluations and is in use at Carnegie Mellon University. The Intelligent Maintenance Training System (IMTS), developed by the University of Southern California and Search Technology, will soon leave the laboratory to be experimentally implemented into a Navy helicopter training course.

Table 1. Examples of Expert Systems for Training

SYSTEM	CONTENT	DEVELOPERS/DATES
SOPHIE	Electronics	Brown, Burton, & deKleer, 1974-1984
SCHOLAR	Geography	Carbonell & Collins, 1968-1972
LISP Tutor	LISP Programming	Anderson, 1983-1986
Geometry Tutor	Geometry	Anderson, 1983-1986
STEAMER	Ship Steamplant Operation	Hollan, Hutchins & Weitzman, 1978-1986
BUGGY/DEBUGGY	Math	Burton, 1979-1981
GUIDON	Medicine	Clancey, 1980-1986
WEST	Solving a Math Game	Burton & Brown, 1978-1984
QUEST	Electronics	White & Fredriksen, 1984-1988
IMTS	Navy Helicopter	Behavioral Technology & Search Technology, 1984-1988

2.2 Capitalizing on Previous Intelligent Tutoring Systems

The history of ITSs for simulation, if not the history of all ITSs, begins with the SOPHisticated Instructional Environment (SOPHIE). SOPHIE (Brown, Burton, & de Kleer, 1982) was designed to be a reactive learning environment in which students could create and test hypotheses and receive pertinent feedback. The emphasis was on building a safe "lab" environment for electronics troubleshooting. Critics have suggested that SOPHIE was limited since it simulated only one problem with one part at a time, while in the real world multiple problems occur simultaneously. Instructionally, however, it was necessary to isolate one problem at a time to avoid confusing the student. As will become clear later, we made the same decision for the MITT project.

SOPHIE was instructionally useful for several reasons. The students became familiar with the functional relationships among electronic components in the system. The students also learned how to solve problems in a systematic and efficient manner. Later versions of SOPHIE included an articulate expert that reasoned in human-like fashion. This capability enhanced the instructional capability of SOPHIE such that students could see step by step how an expert would solve the very problem they just experienced. To incorporate the instructional ability of SOPHIE, the MITT project instantiates two experts, a functional expert and a procedural expert. Each provides the student with clear information in each expert's domain.

STEAMER was an attempt to create a manipulable simulation of a steam propulsion plant of a ship. The engineers who run these ships need years of experience before they acquire the necessary expertise. Williams, Hollan, and Stevens (1981) saw the main training problem as creating a mental model of the steam plant and understanding the related engineering principles. Ancillary to this training task was the formation of procedures for running the plant. The formation of the mental model is a pragmatic approach to instruction since learning all of the necessary procedures by rote is impractical.

The STEAMER project became a pioneering effort in the development of interactive graphic simulations. The developers were the first to use a mouse interface, as well as general purpose graphic and design tools. STEAMER was designed to employ these tools in such a way that instructors could use them and extend the tools' instructional capabilities. One of the observations in the development of the STEAMER intelligent tutor was that abstractions were very useful not only in devices but also with procedures (Stevens & Roberts, 1983). The rationale was that relatively few abstract devices and procedures needed to be used to generate pedagogically sound explanations. Mental models are also abstractions and they are what experts actually use (Hollan,

Hutchins, & Weitzman, 1984). For this reason, the level of abstraction chosen for the MITT project was one that corresponds to experts' mental models and not to physical shapes, sizes or other trivial characteristics.

Qualitative Understanding of Electrical System Troubleshooting (QUEST) is a fairly recent project related to both SOPHIE and STEAMER. While STEAMER was an attempt to increase the conceptual fidelity of the simulation by improving the granularity of the interface, QUEST (White & Fredriksen, 1986) improved the fidelity in the simulation's internal representation as well. Its internal representation was a qualitative model instead of a quantitative model such as the one used in STEAMER. It was similar to SOPHIE both in its domain and instructional philosophy. Like SOPHIE, QUEST was designed to provide to the student a reactive environment in which to solve circuit problems. It was different from SOPHIE in that it was flexible enough to allow the student to build or modify circuits while still receiving explanations about parts and procedures.

The MITT project capitalizes on the lessons learned from all of these systems. The MITT system incorporates the current thinking about mental models and also contains explanation facilities comparable to the most advanced simulations. The research from projects such as SOPHIE, STEAMER, QUEST and other systems, as well as our own experience described in Section 2.3, permitted the developers to build MITT in only 6 months.

2.3 Research on Computer-Based Instruction

This section briefly describes the more than 10 years of research that Johnson, Hunt, and Rouse have conducted in computer-based simulation for diagnostic training. This work formed the evolving foundation for MITT. The Framework for Aiding the Understanding of Logical Troubleshooting (FAULT) simulation described below is at the heart of the MITT system.

Our diagnostic training research (Hunt & Rouse, 1981; Johnson, 1981, 1987; Johnson & Rouse, 1981; Rouse & Hunt, 1984) is characterized by an evolving set of computer simulations and extensive experimental and real-world evaluations. This R&D has been in diverse domains such as automotive and aviation mechanics, communication/electronics and nuclear safety systems. Our work began as an attempt to understand how humans gather and process information in problem solving situations. This led to the consideration of the effects of training on problem solving behavior. A variety of training concepts were developed and tested in laboratory and in field tests. Our focus has been and remains on aspects of training rather than job aiding.

The two diagnostic training simulations that emerged from this research were Troubleshooting by Application of Structural Knowledge (TASK) (Rouse, 1979) and FAULT (Hunt, 1979).

The TASK simulation involves a context-free network of parts having from 9 to 49 components. The troubleshooter must search the network for a failure by making observations among the parts. Variations on TASK permitted the study of computer-based aiding during training (Rouse, 1979), feedback (Pellegrino, 1979), performance prediction, (Rouse & Rouse, 1982) and transfer of training (Hunt, 1979; Johnson, 1981). The TASK simulation is not part of MITT, but it has most certainly affected our understanding of human problem solving training.

The second simulation, FAULT, is a context-specific derivation of TASK. It uses a hard copy functional flow diagram with an on-line display for student options, test results and feedback. The simulation permits the user to engage in the same information processing that would take place during real equipment troubleshooting. This includes actions such as checking instruments, obtaining symptomatic reports from an operator, performing both simple and complex diagnostic tests and replacing parts.

The FAULT simulations developed prior to MITT included a limited degree of intelligence that provided student advice and feedback. The intelligent features are listed below:

- o Tracks student progress toward problem solution
  - Record all student actions
  - Calculate the implications of each test result
- o Provides immediate feedback on student error
  - Errors of inference
  - Redundant actions
  - Explanation of why the action was an error
- o Provides advice
  - Identify actions with maximum information gain
  - Identify actions with maximum information gain/cost
  - Identify easy tests
  - Reduce feasible set of failures based on test actions

FAULT consists of two parts, an inference engine and a knowledge base. Current AI convention suggests that, in general, the inference engine should be limited to very simple reasoning processes. The complex relationships inherent to the domain of interest should reside in the knowledge base. Systems built in this fashion have the advantage of being modular and infinitely expandable. This is a necessity in domains such as medicine and mineral prospecting where more "knowledge" is continually being uncovered and existing knowledge seldom becomes obsolete. The disadvantage is that a knowledge base

may become very large and unmanageable even for a very modest problem space. Furthermore, completeness of the knowledge base is difficult to assure.

The approach taken by the FAULT developers was to create an inference engine tailored to the task of technical training and fault diagnosis. By doing this, it was possible to create a very robust simulation with a very modest knowledge base. As new systems are developed and old ones become obsolete, this approach minimizes the costs associated with creating new knowledge bases. The disadvantage to this approach is the difficulty of customizing a knowledge base to represent system-specific peculiarities.

For most expert systems, the inference engine is likely to be a smaller part of the program than is the knowledge base, which is composed of many rules. The implications are that each new tutoring system requires a substantial new effort on the part of the knowledge engineers and team of content specialists. While the expert system is considerably more difficult to develop than a FAULT system, it does have the advantage of more context-specific advice in the form of if-then rules. MITT is an attempt to combine these two approaches (see Figure 1). The FAULT-like inference engine makes developing knowledge bases relatively easy. The C Language Integrated Production System (CLIPS) structure permits the inclusion of varying amounts of robust descriptions specifically related to the system users' technical/operational procedures.

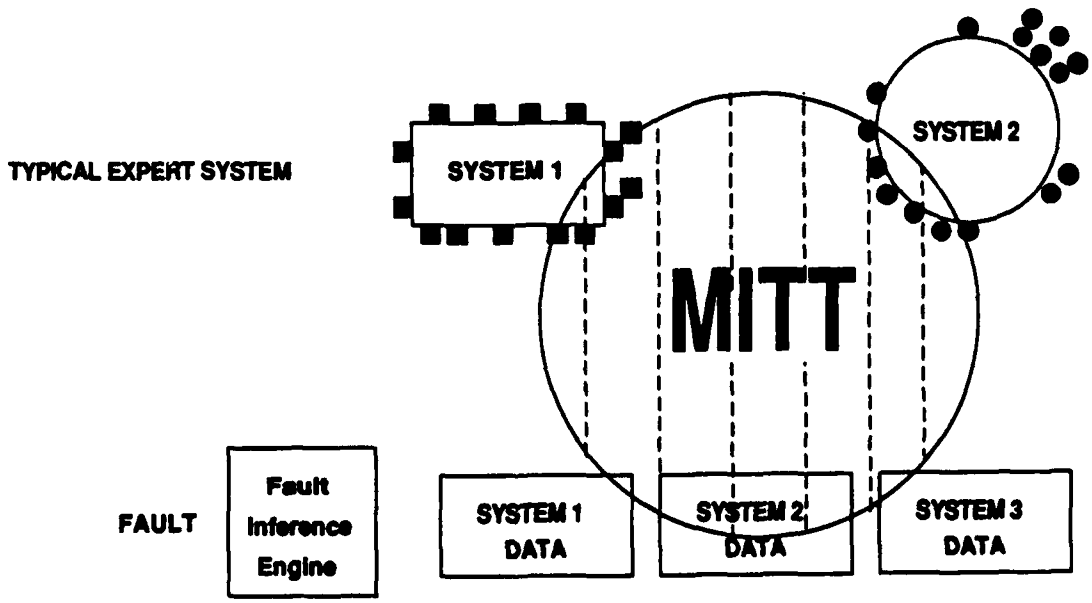


Figure 1. MITT is a Combination of the FAULT and Typical Expert System Development Technology

3.0 THE MITT SYSTEM

The MITT system consists of five modules (see Figure 2). This section describes each of the modules and how each of the modules communicate to form a tutoring environment. MITT provides enhancements to all modules of past FAULT systems. Each of these modules and enhancements is discussed in this section.

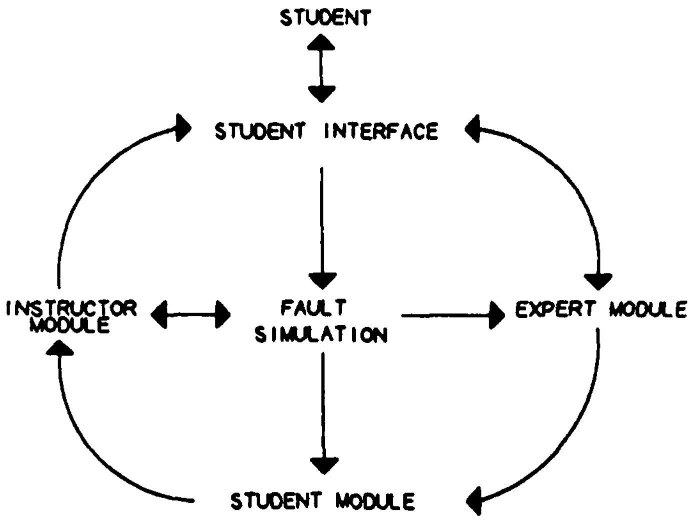


Figure 2. The Modules of MITT

3.1 The FAULT Simulation Module

The FAULT simulation module serves as the heart of MITT. Perhaps the greatest strength of the FAULT simulation, see Section 2.3, is the simplicity of the system representation. Each part in the system is represented as a node in a network. Lines connecting the nodes represent input/output (I/O) relationships among parts. During the design and development of MITT, we explored the feasibility of assigning a variety of relationships (e.g., mechanical, electrical, hydraulic) to the I/O relationships. We found that the mixture of relationships was confusing to the student. The simple, single functional relationship seems to permit the learner to create more quickly a mental model of the technical system.

3.2 The Procedural/Functional Expert Module

MITT's expert module maintains representations of the system in two ways. First, there is a procedural expert that maps symptoms, such as annunciator lights and instrument indications, to suspect systems and components with collections of if-then rules. The second representation, a functional expert, uses one or more connectivity matrices representing various functional relationships among parts.

3.2.1 The Procedural Expert

The Procedural Expert (PE) is a series of rule-based statements that contain information, in the form of advice, about specific NASA troubleshooting procedures for the fuel cell. At any stage in the diagnosis of a malfunction, the student can request procedural advice and receive specific information about what to do next. The PE makes decisions and provides advice according to the gauges, controls or panels the student has seen and the order in which they have been seen. This expert works only when requested by the student.

The PE is divided into submodules that correspond to the malfunctions. Since memory is at a premium only one module is active at any one time. The demand-driven PE works as a menu option from the main MITT menu. The procedural advice is a statement of the suggested next step to take in the diagnosis of a malfunction. After the advice appears, the student can choose to use any of the other menu options.

PE development required an analysis of the NASA troubleshooting procedures. Such procedures are available for all systems and are usually very thorough. These procedures were broken down into steps or actions for the student. Each action was equated to reading gauges, controls or panels. Our analysis had to ensure that the student had some means to accomplish a given NASA procedure within the constraints of the simulation.

Each statement of advice is linked to a logical test. If the orbiter primary annunciator panel (F7) is seen and the front gauge panel (F9) or cathode-ray tube (CRT) display System Summary 1 is seen, then a particular advice statement follows. All of the advice statements for the six initial malfunctions were incorporated into about 60 rules. Additional procedural intelligence can be incorporated merely by adding to this rule base.

3.2.2 The Functional Expert

The Functional Expert (FE) is based on the connectivity among the system components. For example, if part A depends on part B and part B has failed, then part A will be adversely affected. Also, part A will adversely affect parts which depend on it. This functional understanding permits the expert to calculate how failures propagate through the system following the functional topography.

The student is able to interact with this expert by requesting FE advice while working on a problem. This expert is capable of responding with assistance based on the student's previous actions. The expert will also communicate with the Student Module as discussed in Section 3.3.

3.2.3 Integration of the Experts

It is necessary that the experts talk to each other. This capability was accomplished for MITT by the creation of an equivalency table between gauge readings and topographic tests. For example, a reading of the $O2/H2$ flow on a gauge may be the same as a test between two parts of the functional schematic. As far as MITT is concerned, they are the same. All of the tests were translated to gauge readings to match the format requirement needed by the PE.

3.3 The Student Module

The Student Module is predominately a tally of the student's actions throughout the simulation thus creating a model of the student. This model is updated by the FAULT Simulation Module and the Expert Module. The student model is used by the Instructional Expert to determine when advice is appropriate.

The initial version of MITT has a limited student model that is current only for each problem. The model of the current problem status includes tests the student has performed, the results of the tests and what can be inferred from those tests regarding the current problem. Each additional test performed by the student causes the student model to update the current problem status according to the information gained through the test. The student model keeps track of the number of times the student uses each action from the primary simulation display as well as the number of accesses to each orbiter gauge, annunciator or CRT. The model also keeps track of the type of errors the Functional Expert is noting. The Student Module provides feedback upon completion of each problem (see Figure 3).

Congratulations! You have corrected the problem.

Number of students completing this problem:	3
Number of students who quit before solving:	1
Number of students who received a time out:	0

	Yours	Average
Number of minutes for repair:	14.2	7.30
Number of errors made:	3	0.50
Number of displays accessed:	17	8.01
Number of times procedural advice used:	4	.83
Number of times functional advice used:	6	1.36

Figure 3. MITT Feedback at Problem Completion

We believe that a historical student model should be added to MITT in the next version. The historical model within the Student Module would include summary information regarding student performance for all problems seen by the student. This information would include, for example, the frequency of use of each available test type, the types and number of errors committed and the quality or effectiveness of the tests chosen. These data will allow the model to interpret errors such as failure to make inferences, failure to make proper inferences, failure to recognize significant relationships and so on. This information should be used as a profile to describe the student's mastery of the material for multiple problems.

3.4 The Instructional Expert Module

The Instructional Expert (IE) module is a rule-based routine that pinpoints certain student errors and intervenes immediately after they occur. At this stage of the implementation, the errors detected by the IE are neither subtle nor exhaustive. The types of errors detected now include student actions that result from misunderstanding the MITT system or simple troubleshooting procedures. The advantage of this initial approach is that the IE provides somewhat generic advice that promises to be effective for current and future simulations.

The IE was designed to help guide the student to the most appropriate segment of the MITT system. Instead of redundantly explaining something, the IE suggests where the student should look for more information. In this sense, the IE actually works more like a reference librarian, directing the student instead of teaching. The advantage of this approach is that only those who need the reference material need choose to use it.

The entire MITT system is designed to deliver instruction so simply that a novice can use the system. The system is replete with helps, advice and information about the fuel cell. Since it is also student-driven, the initiative lies therein. Only in exceptional circumstances should the IE intervene to redirect students from their own exploration paths.

Since there is a limit to the amount of memory available for MITT, certain decisions were made to use the memory to its fullest. One of these decisions was to implement the IE in the C language. By doing this, the IE could intervene when necessary without having a CLIPS program running in background mode and draining memory.

3.5 The Student Interface Module

Although each module of an ITS is critically important, it is the user interface that ultimately delivers the instruction to the user. The interface must allow easy interaction with the simulation. Also, the outputs of the Procedural/Functional Expert, Student, and Instructional Expert modules must be well integrated with the interface in order to be effective and unobtrusive.

The primary user input device is a mouse, although a keyboard may also be used in all cases. The cursor, controlled by either the mouse or the keyboard, can be moved within each menu to select student diagnostic options. On certain screens, extensive data are available. In order for the simulation to know precisely what data the student wants, it is necessary for the user to click on the *XXX* area (see column 3 in Figure 4) to obtain the information. An example of this type of display is shown in Figure 4.

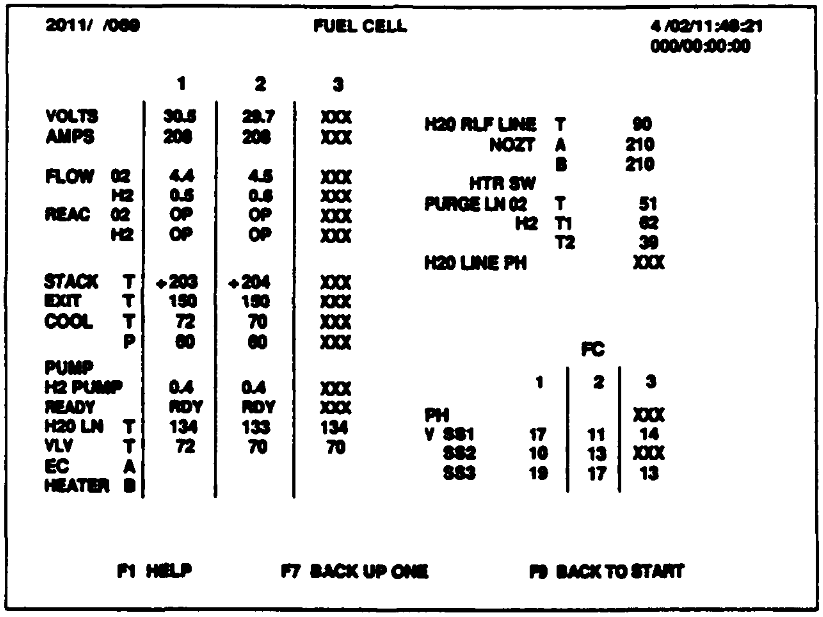


Figure 4. CRT Display Requiring Specific Information Request Areas

At the outset of this project, we considered placing the hard copy functional flow diagrams on the screen. The size of the diagrams, with necessary labeling, exceeded the graphics capability of the technology commonly available to NASA and Air Force training environments. The use of hard copy presents no interface difficulty since hard copy schematic diagrams are typically used for most troubleshooting. Furthermore, thousands of FAULT users in numerous domains have used the hard copy functional flow diagrams with relative ease.

Throughout the MITT development, our goal was to ensure that use of the software was relatively intuitive. There is an easily understood, online description of "how to use the simulation" offered at the front of the program. A hard copy of the user's manual is included as Appendix A of this paper.

3.6 Hardware and Software

The hardware system for development and delivery is an IBM-AT or an IBM-compatible system. The system requires 640 Kb of random access memory (RAM) and a hard disk. Presently, a Color Graphics Adapter (CGA) card and color monitor are also needed. A mouse is recommended, but optional.

The rationale for using the IBM-AT is straightforward. First, the equipment is affordable and likely to be found in most research laboratories and, more importantly, training installations. When a group makes the decision to use MITT, they will not have to purchase expensive, dedicated AI machines. Second, the machine has all the necessary speed, storage, hardware and software support, and other capabilities to deliver the required level of intelligent tutoring. In addition, the IBM-AT has readily available off-the-shelf peripherals for interface to video disk and other computers including mainframes that may be of use for future expansion.

MITT is written in C. The program processes rules for portions of the Instructional Expert, Student and Procedural/Functional Expert modules without using LISP. The NASA expert system CLIPS was used.

3.6.1 CLIPS as an Expert System Tool

For our purposes, CLIPS was a convenient tool for several reasons. It could be embedded into our existing C code, it has an inference engine built in, the rules are simple to create and CLIPS itself is easy to learn.

The disadvantage was that we did not have a reliable, compiled version of CLIPS. When the compiled version of CLIPS becomes more reliable, the speed of running procedural advice should improve quality. The noncompiled CLIPS, therefore, caused a considerable delay for the response to the student and made it impossible to incrementally update the Student Module. The reference manual is opaque with regard to passing strings and variables generated by CLIPS to C. This was a severe limitation for MITT, which is based upon message passing among several experts, only one of which is mute.

The inability to run the CLIPS executable files simultaneously with the C executables placed restrictions on the procedural advice of CLIPS. The biggest restriction

was that only one advice rule could fire per run. Since it was unclear how to pass messages from CLIPS to runtime C, all of the information passed was from C to CLIPS.

4.0 DEVELOPMENT AND FORMATIVE EVALUATION

This project started with the ambitious goal of designing, developing and demonstrating a microcomputer-based ITS, all within a 6-month timeframe. The goal was accomplished with careful planning, reliance on past CBI experience and ongoing formative evaluation. This section describes the development process.

4.1 Selecting the Instructional Domain

The short duration of this project meant that we had to quickly identify an appropriate technical domain and elicit the cooperation of technical experts in that domain. Throughout the conceptualization and development, it was important to remain in close communication with those experts.

Selecting an instructional domain was simplified by AFHRL because they wanted to use a space application and work with NASA at Johnson Space Center (JSC). We became associated with the NASA personnel who develop and deliver training for astronauts and flight controllers. This group recognized the need for an ITS and showed a willingness to cooperate. They selected the orbiter fuel cell subsystem as an appropriate domain for the demonstration. The space shuttle fuel cell subsystem met the criteria described by Johnson (1988). The fuel cell system turned out to be a reasonable first challenge for MITT.

In a project such as this one, where the work must be accomplished in a very short time, a change of subject-matter experts (SMEs) could have an adverse effect on the schedule. NASA did change experts halfway through the project, but fortunately, this did not hurt the schedule. Both SMEs contributed a great amount of technical expertise and instructional enhancements while remaining responsive to the short project schedule. Thus, we were able to continue the project without any schedule slips.

4.2 Evaluation

Evaluation can be divided into two stages, formative and summative. The formative evaluation takes place during software design and development. Summative evaluation refers to the measurement of the software's value following development. For this short effort, the formative evaluation is most important.

4.2.1 Formative Evaluation

The primary goal of formative evaluation is to keep the potential user informed of the ongoing development and expected final product. Formative evaluation permits the developers, SMEs and prospective users to be constantly informed and able to make real-time changes in the design. Formative evaluation prevents the comment that "it's too late to change that now."

The majority of this project's formative evaluation was accomplished through ongoing interaction between the MITT developers and personnel from NASA and AFHRL. Slightly over halfway through the project, we held a prototype review meeting. At this meeting, we demonstrated screen displays and interfaces, and then depicted the general diagnostic ITS scenario that the MITT program would deliver. Acting on feedback from that prototype review, we made appropriate modifications and proceeded to complete the MITT program.

During the development of the program, we followed a proven software evaluation plan (Maddox & Johnson, 1986). The plan ensured that software evaluation was performed using a three-step process: measuring compatibility, measuring understandability and measuring effectiveness. Compatibility refers to the extent to which the user is able to see the computer displays and reach the keyboard and other pointing devices. Displays must be legible and all colors must be easily discriminable. Understandability refers to the output from the ITS and the input it requires. The I/O must match the user's prior knowledge and training. Users must be able to understand what the system is telling them. Effectiveness can be equated to summative evaluation.

While we were able to completely assess compatibility at the development site, certain aspects of understandability were measured using the demonstration/evaluation at JSC. Based on our observations and comments from users, the MITT software is both compatible with and understood by NASA users.

4.2.2 Demonstration/Evaluation: A Measure of User Acceptance

The final demonstration/evaluation took place on 17-18 December 1987 at JSC. In that 2-day period, MITT was used by 17 NASA personnel including astronauts, flight controllers, CBI developers, AI researchers, technical instructors and a training manager.

The user acceptance was overwhelmingly positive. Users enjoyed the approach to diagnostic training. We noticed that the flight controllers were more comfortable with the approach than were the astronauts. By the nature of their jobs, the flight controllers collect and analyze data to make decisions. Astronauts, on the other hand, are more inclined to press buttons and throw switches. The MITT program seems to be more

compatible to the flight controllers' job tasks.

At the demonstration summary meeting, NASA was very positive about MITT and was anxious to continue working with AFHRL. NASA was concerned about the incompatibility between the IBM EGA graphics and the AT&T graphics, and felt it was important that the subsequent work be compatible with NASA's prime training equipment. As a result, all MITT graphics were converted to CGA to insure compatibility with AFHRL and NASA microcomputer hardware.

4.2.3 Comments About Summative Evaluation

A summative evaluation should assess the extent to which MITT contributes to a change in the learner's knowledge, skill and perhaps attitude as a fuel cell expert. There are a number of ways to perform such an evaluation. These evaluations would include transfer-of-training assessments with written and/or performance tests. The performance tests could be conducted with the variety of space shuttle cockpit simulators available at Johnson Space Center. MITT could also be used to assess performance change.

Most of these evaluation methods could be implemented at JSC with minimal disruption of instruction. We suggest that the flight controllers would gain the most from MITT. The population of flight controllers is much larger than the number of astronauts. More students (controllers) would add to the power of any statistical analysis. Prior to any such evaluation, MITT must undergo a few changes as outlined in Section 4.2.3.1.

4.2.3.1 MITT Modifications for Summative Evaluation. MITT was received positively in its 2-day demonstration at JSC. However, if an extensive transfer-of-training summative evaluation is to take place, the following changes must be made.

First, the software should be modified to include a data collection routine. This software would maintain a transaction file of all user interactions with MITT. Such data could be used to determine such factors as learning curves, user-simulation interface trouble spots, actions commonly used or not used, as examples. The evaluation plan should drive the design of data collection software.

The second MITT modification for substantive evaluation is the addition of extra fuel cell failure scenarios. We suggest that the current set of 6 be at least doubled. With a total of 12 failures, most students could use the simulation for 2 to 3 hours.

The changes above are the only ones that are critical for the summative evaluation; however, additional enhancements are possible.

5.0 SUMMARY AND SUGGESTIONS FOR CONTINUED RESEARCH

The goal of this project was to design, develop and demonstrate a fully operational ITS, in a space application, within 6 months. We met that goal with an ITS that was perceived by NASA to have potential value for space training. While we met the goal, we believe that MITT is merely a first step in this microcomputer ITS research.

We believe that each module--the Student, Instructional Expert and Procedural/Functional Experts--can now be enhanced. Perhaps the next step would be to enhance the Student Module by adding student historical data that could fire new and improved rules for the Instructional Expert. The Instructional Expert as well as the Procedural/Functional Expert could also be embellished to learn from user actions. Since MITT is modularized, it is possible to modify each of these experts somewhat independently of the others.

While each of the above ideas should be elaborated upon and then pursued, the most logical next step would be to develop a MITT simulation for a second instructional domain. This would serve to demonstrate the relative generalizability of MITT. Furthermore, it would provide a means to accurately estimate the cost of MITT application development.

Another step in the MITT's evolution is the development of an authoring system to permit technical experts to build MITT without writing computer code. To do this, a detailed functional specification of such a system should first be prepared. This specification would serve as a guide for follow-on AFHRL-sponsored efforts.

6.0 REFERENCES

Anderson, J.R., Boyle, C.F., & Yost, G. (1985). The geometry tutor. In A. Joshi (Ed.), *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 1-7). Los Altos, CA: Morgan Kaufmann.

Anderson, J.R., & Skwarecki, E. (1986). The automated tutoring of introductory computer programming. *Communications of the ACM, 29*, 842-849.

Brown, J.S., Burton, R.R., & de Kleer, J. (1982) Pedagogical, natural language, and knowledge engineering techniques in SOPHIE I, II and III. In D.H. Sleeman and J.S. Brown (Eds.), *Intelligent Tutoring Systems* (pp. 227-282). New York: Academic Press.

Burton, R.B., & Brown, J.S. (1982). An investigation of computer coaching for informal learning activities. In D. Sleeman & J.S. Brown (Eds.), *Intelligent tutoring systems* (pp. 79-98). New York: Academic Press.

Carbonell, J.R., & Collins, A. (1973). Natural semantics in artificial intelligence. *Proceedings of the Third International Joint Confernece on Artificial Intelligence* (pp. 344-351). Los Altos, CA: Morgan Kaufmann.

Clancey, W.J. (1987). *Knowledge-based tutoring: The GUIDON program*. Cambridge, MA: MIT Press.

Hollan, J.D., Hutchins, E.L., & Weitzman, L. (1984) *STEAMER: An interactive inspectable simulation-based training system*. AI Magazine, *5*(2),15-27.

Hunt, R. (1979). *A study of transfer of training from context-free to context-specific fault diagnosis tasks.* Unpublished master's thesis, University of Illinois: Urbana, IL.

Hunt, R.M., & Rouse, W.B. (1981). Problem solving skills of maintenance trainees in diagnosing faults in simulated power plants. *Human Factors, 23*(3), 317-328.

Johnson, W.B. (1988). Pragmatic considerations in development and implementation of intelligent tutoring systems. In R. J. Richardson and M. Polson (Eds.), *Foundations of intelligent tutoring systems* (pp. 191-207). Hillsdale, NJ: Lawrence Erlbaum Associates.

Johnson, W.B. (1987). Development and evaluation of simulation-oriented computer-based instruction for diagnostic training. In W.B. Rouse (Ed.), *Advances in man-machine systems research: Vol. 3* (pp. 99-125). Greenwich, CT: JAI Press.

Johnson, W.B. (1981). Computer simulations for fault diagnosis training: An empirical study of learning from simulation to live system performance. (Doctoral Dissertation, University of Illinois, 1980), *Dissertation Abstracts International, 41*(11). 4625-A. (University Microfilms No. 8108555).

Johnson, W.B., & Rouse, W.B. (1981). Analysis and classification of human errors in troubleshooting live aircraft power plants. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-12*,(3), 389-393.

Maddox, M.E., & Johnson, W.B. (1986). "Can you see it? Can you understand it? Does it work? An evaluation plan for computer-based instruction. *Conference Proceedings: Advances in Human Factors in Nuclear Power Systems*. LaGrange IL: American Nuclear Society.

Maddox, M.E., Johnson, W.B., & Frey, P.R. (1986). *Diagnostic training for nuclear power plant personnel. Vol. 2: Implementation and evaluation* (EPRI NP-3829). Palo Alto, CA: Electric Power Research Institute.

Pellegrino, S.J. (1979). *Modeling test sequences chosen by humans in fault diagnosis tasks.* Unpublished master's thesis, University of Illinois: Urbana, IL.

Psotka, J., Massey, R., & Mutter, S. (Eds.). (1988). *Intelligent tutoring systems: Lessons learned.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Richardson, R.J., & Polson, M. (Eds.). (1988). *Air Force Human Resources Laboratory Research planning forum for intelligent tutoring systems.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Rouse, S.H., & Rouse, W.B. (1982). Cognitive style as a correlate of human problem solving performance in fault diagnosis tasks. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-12*(5), 649-652.

Rouse, W.B. (1979). Problem solving performance of first semester maintenance trainees in two fault diagnosis tasks. *Human Factors, 21*(5), 611-618.

Rouse, W.B., & Hunt, R.M. (1984). Human problem solving in fault diagnosis tasks. In W.B. Rouse (Ed.), *Advances in man-machine systems research: Vol. 1.* Greenwich, CT: JAI Press.

Rouse, W.B., Rouse S.H. & Pellegrino, S.J. (1980). A rule-based model of human problem solving performance in fault diagnosis tasks. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-10*(7), 366-376.

Stevens, A.L., & Roberts, B. (1983). Quantitative and qualitative simulation in computer-based training. *Journal of Computer-Based Instruction, 10*(1), 16-19.

Towne, D.M. (1988). The generalized maintenance trainer: Evolution and revolution. In W.B. Rouse (Ed.), *Advances in man-machine systems research: Vol. 3* (pp. 1-63). Greenwich, CT: JAI Press.

Wenger, E. (1987). *Artificial intelligence and tutoring systems.* Los Altos, CA: Morgan Kaufmann.

White, B.Y., & Fredriksen, J.R. (1986). Qualitative models and intelligent learning environments. In R. Lawler and M. Yazdani M. (Eds.), *AI and education: Learning environments and intelligent tutoring systems* (pp. 281-304). Norwood, NJ: Ablex Publishing.

Williams, M.D., Hollan, J.D., & Stevens, A.L. (1981). An overview of STEAMER: An advanced computer-assisted instruction system for propulsion engineering. *Behavior Research Methods and Instrumentation, 13*(2), 85-90.

Appendix A

Student Instructions for Running MITT Fuel Cell Simulation

STUDENT INSTRUCTIONS
FOR
RUNNING MITT FUEL CELL SIMULATION

Microcomputer Intelligence For Technical Training

NASA

AFHRL

Search Technology, Inc.

Search Technology, Inc.
Suite 200
4725 Peachtree Corners Circle
Norcross, Georgia 30092
(404) 441-1457

To start the MITT Simulation Program*,

1. Turn on the computer.

2. Use DOS commands to create directory "MITT". Copy diskettes into MITT directory (this only needs to be done one time).

3. At the DOS prompt C>, type CD MITT and press "Enter."

4. At C> prompt, type MITT and press "Enter."

5. Follow instructions given by the simulation.

   * It is essential that your DOS \CONFIG.SYS file contains the line, *Files=20*

   * You must have a minimum of 590k of RAM available

For technical assistance call:

Jeffrey E. Norton
William B. Johnson
Phillip C. Duncan

(404) 441-1457

NOTICE

This Courseware should be used *ONLY* for training purposes. The schematics, drawings, diagrams, instrument layouts and specific instrument readings are idealized and *DO NOT* necessarily represent actual conditions or values.

1.0 Purpose of the Simulation

A SIMULATION is a model of real life. For example, video arcades have simulated race cars, fighter jets, and tanks. Some simulations are exactly like real life, in that they contain the aspects of system appearance displays, noise, motion, etc. The full motion shuttle simulator is an example of simulation designed to be identical to the real equipment.

The MITT Simulation is not exactly like real life. It does not include motion, noise, and full visual systems of a flight simulator. However, the simulation does contain the information to permit you to think through the same steps included in real equipment troubleshooting. For example, you can check panel annunciators and gauges. The simulation assumes that you know how to locate and interpret instrumentation/displays thus permitting you to get right down to system troubleshooting and repair.

The PURPOSE OF THE SIMULATION, therefore, is to permit you to apply your knowledge to system troubleshooting. If you mix the use of this simulation with classroom lectures, other simulator training, and ancillary reading and study, you will be ready to diagnose problems when they occur.

1.1 The Computer

The computer software is prepared for an IBM-AT with a color monitor and color graphics adapter card (CGA). The system requires a hard disk and 640 Kb RAM. The MITT directory will require 1.5 megabytes on the hard disk.

1.1.1 Using the Computer

Your input to the computer will be with the keyboard and/or the mouse. The keyboard is described in Section 1.1.2. The mouse is described in detail in Section 1.1.3.

1.1.2 You will need to type only *individual* letters and numbers.

In addition, you may use the key labeled "<-" to delete any typing input errors. In most cases, you will have to press the "**Enter**" key to tell the computer to read what you have typed.

The keyboard can also be used to move the crosslike (+) pointer around the screen when a mouse is not available. The arrows on the right keypad will move the cursor up, down, and sideways. The function (F) keys will provide "Help" for each page, F7 will permit you to back-up to the previous page, F9 will bring you to the highest menu, and F10 will permit you to select the option to which the cross-like (+) pointer is pointing.

1.1.3 The Mouse

The MOUSE allows you to communicate with the computer without using the keyboard. Slide the mouse across the tabletop, to move the pointer on the screen.

Depending on the model used, there will be either 2 or 3 buttons on the mouse. The mouse buttons serve the same function throughout the simulation. The left button on the mouse is the same as the "**ENTER**" key; it is also called the "pick" button. Usually, you will move the pointer to your menu selection then press the pick button. The center button is a backup key, in that it will permit you to go back to the previous page. You can press the keyboard "**ENTER**" to accomplish the same function as the mouse pick button. The right button will lead to specific instructions or help. When a two-button mouse is used, help can be obtained using the F1 key.

A program to install the mouse is included on the MITT diskettes. The program, **FCINSTAL**, will let you select among keyboard or type of mouse. You must load the mouse software that comes with your mouse.

1.2 To Begin

You will need this manual to use the simulation. While most directions are available on the computer, the manual must be used for additional instructions. It also includes the important hardcopy subsystem displays that will be necessary to use the simulation. These system displays are found at the back of the manual.

To start the simulation, turn to the first page of this manual. You will find brief instructions on how to load the software and run the simulation. If the program has been installed for your machine, type **MITT** to begin.

The first display of the Fuel Cell (FC) simulation is shown in Fig. A-1. To begin, press either the "**ENTER**" key or the "pick" button on your mouse.

Figure A-2 shows the second display. From this display, you may gather information about the simulation or begin a problem. This option will permit you to learn about the mouse and keyboard, to learn how to use the simulation, to look at FC system schematics and learn or review subsystem components and their functions, or to begin problem solving. Do not choose option 4 until you are familiar with the rules of the simulation. Once you select option 4, a 20-minute countdown clock starts timing your troubleshooting performance. If you do not solve the problem within 20 minutes, the simulation will "time-out" and give you the answer. To check your progress on the 20-minute countdown clock, use the F1 - Help button.

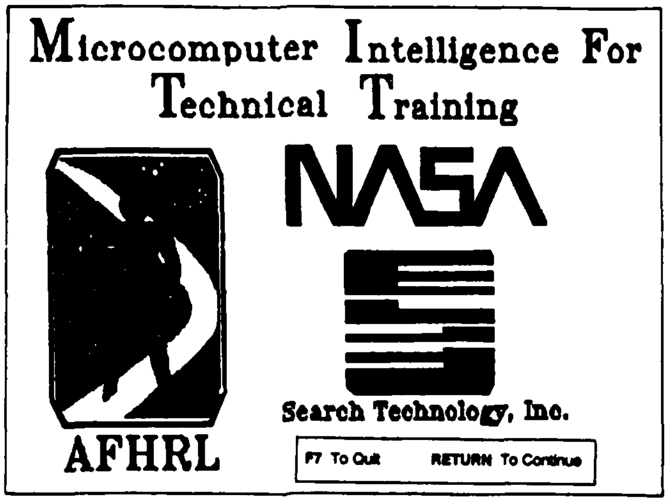


Figure A-1. First Display on FC Simulation

1. Mouse and Keyboard Information
2. Simulation Information
3. Fuel Cell Information
4. Start or Resume a Problem

Type number or use mouse to select option

F1 Help F7 Back Up One F9 Back To Start

Figure A-2. Main List of Simulation Options.

1.2.1 Mouse and Keyboard Information

This mouse and keyboard information tells you how to use the mouse if one is installed. It also provides specific instructions on how to use special keys on the keyboard when a mouse is not available.

1.2.2 Simulation Information

This information can be accessed before or during a problem. During a problem when you return to look at simulation information, the 20-minute countdown clock will stop until you leave this section.

1.2.3 System Information

Here the simulation provides information about the FC system being simulated. The information presented here describes normal operation and does not change with each problem. You can access this information before and during a problem. When you return to this section of the simulation during a problem, the 20-minute clock will stop until you resume the problem.

1.2.4 To Start or Resume a Problem

The next section of this manual describes the options available once you begin troubleshooting.

2.0 System Troubleshooting with a Framework for Logical Troubleshooting

This simulation is designed to help you think about the functional interconnection of various subsystem components for the specific technical system and learn the procedures for diagnosing problems. In addition, continued application of the logical troubleshooting applied here has the potential to sharpen your troubleshooting skills on all technical systems. As you will see, the simulation contains information and advice to help you approach a technical problem in a logical manner.

The simulation requires the use of one of the hardcopy subsystem diagrams included at the back of this manual. These diagrams will help you think about the parts and functions of the various subsystems. The diagrams are meant to depict the functional (but not necessarily physical) relationships among the parts.

On the diagrams, function flows from left to right. In order for a part to have acceptable function flow (output) from its right side, the part itself <u>and</u> all inputs must also be acceptable. The simulation permits you to test that flow to find the failed component.

It is important that you understand some terminology that will be used in the program. When one part connects to another, this implies a direct connection (i.e., the output of one part is an input to the other). When a part reaches another part is means that there is a path between the two parts, but the path may involve other parts in between. For example, look at the parts in Fig. A-3. The parts are numbered 1, 2, and 3. Part 1 connects to part 2 and part 2 connects to part 3. Further, part 1 reaches both parts 2 and 3 whereas part 2 reaches only part 3. Part 3 does not connect to or reach any other part.

Remembering these relationships will help you solve the simulation problems.

- o Part 1 connects to part 2.
- o Part 1 reaches part 2.
- o Part 2 connects to part 3.
- o Part 2 reaches part 3.
- o Part 1 does not connect to part 3, but it does reach part 3.

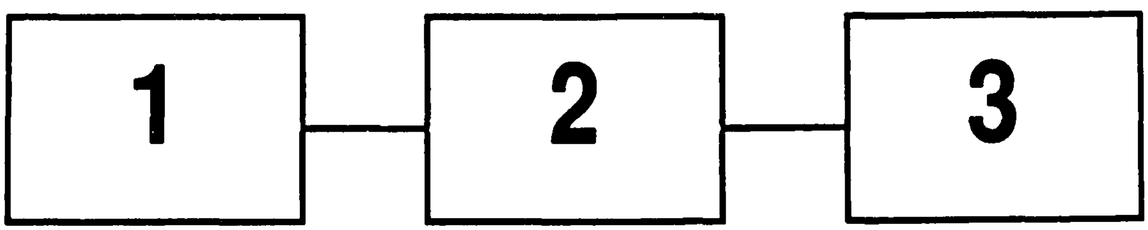


Figure A-3. Relationships Among Three Parts

2.1 Troubleshooting Performance Goal

Your goal in the simulation should be the same as your real equipment goal. That goal is:

1. Determine the System Status
2. If there is a malfunction, find it

The performance measures are:

1. Real time to solution (measured in minutes since starting the problem)
2. Number of testing or replacement errors
3. Number of displays accessed
4. Number of accesses to expert advice

2.2 The Primary Display

Figure A-4 shows the primary display for the majority of actions. The first option permits you to select a subsystem. The available subsystems for the Space Shuttle Fuel Cell are:

Thermal Control
Cryogenic Subsystem

Once you choose a subsystem, you must choose the appropriate subsystem diagram from the laminated cards in the back of this manual. You may change subsystems within a single problem.

ELECTRICAL POWER SYSTEM: Fuel Cell System

FUEL CELL MENU	ACTIONS
GAUGE READINGS . . . . G INFORMATION ON TESTS . . . . Ix g DESCRIPTION OF PARTS . . . . Dx TEST BETWEEN PARTS . . . . . Tx g MCC ANALYSIS . . . . Mx REMAINING PARTS . . . . R FUNCTIONAL ADVICE . . . . . F PROCEDURAL ADVICE . . . . . P ANSWER . . . . Ax QUIT . . . . Q	Your Action > _
Ready for command.	

F1 Help F7 Back Up One

Figure A-4. The Primary Display

2.3 Options

At the top of the screen the system name is displayed. The right side provides a running account of your actions. If this page fills, you can view past actions by using the "page up" key. "Page down" will return you to your current actions.

On the left side of the primary display is the box labeled "Options." Each of these options will be described in detail later. Your input is displayed in the "Your Action" box as you type it. The bottom box is dedicated to providing information and feedback throughout the simulation.

You can select options with the keyboard by typing the letter and number, or with the mouse, by moving the pointer to the appropriate line.

2.3.1 Gauge readings

Gauge readings (G) - Type a G, or slide the mouse to G, to bring up a display which allows you to actually read a particular gauge, selector, talkback, circuit breaker, panel, etc. (See Figure A-5).

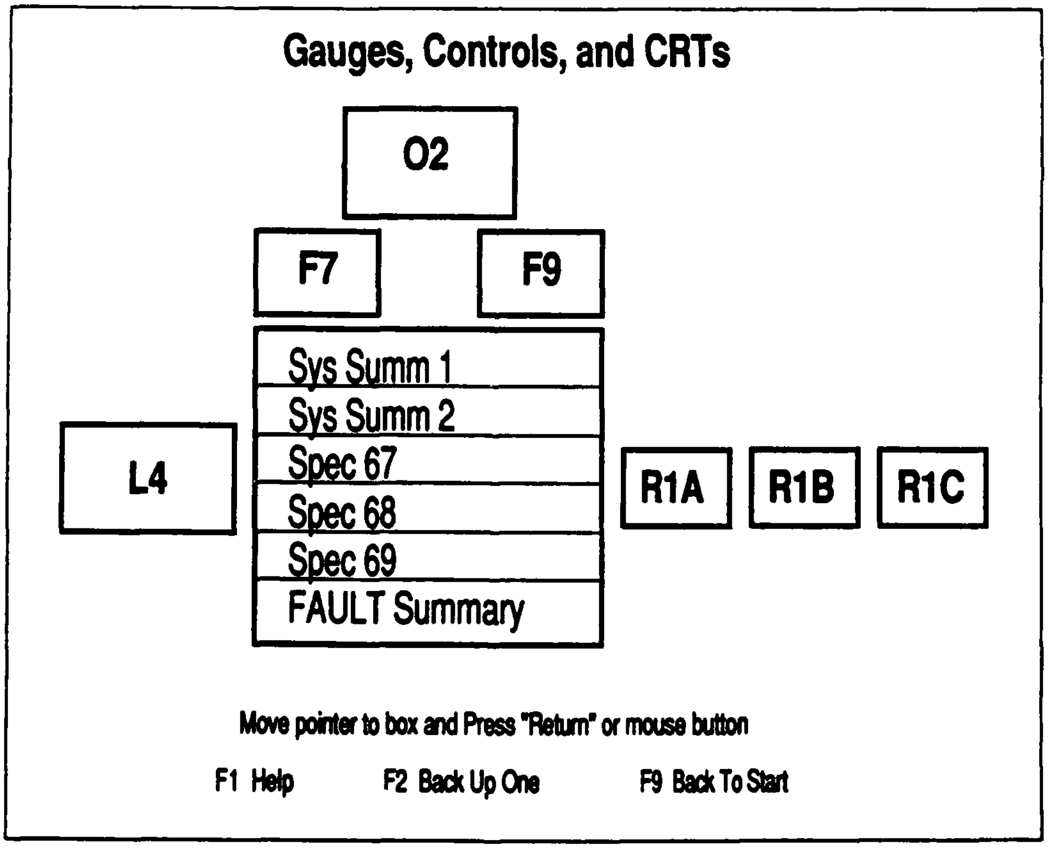


Figure A-5. Gauge Readings Screen

Press the arrow keys on the keyboard to move the location of the cursor over the panel/CRT you wish to view. Then select it by clicking the mouse buttons or press return on the keyboard.

After you select the panel/CRT you wish to view, you may see individual gauges which are hidden. In these cases, you must indicate each gauge on the panel on each each portion of the CRT you wish to read. You can select the gauge or portion to read by using the arrow keys on the keyboard, or by using the mouse and clicking the button. Selecting specific gauges enables the computer to generate a better model or what you have seen and what you haven't seen.

2.3.2 Information on Test

Information on Tests (Ix y) - Ix y will tell you how you would perform a basic observation between part x and part y. Ix will tell you how Mission Control Center (MCC) would perform their analysis of part x.

2.3.3 Description of Parts

Description (Dx) - Type Dx to receive a brief description of part number x.

2.3.4 Test Between Parts

Tx y will show the result of a test between part x and part y. If the result is normal, then component x <u>and</u> every part that reaches x has a normal output and has not failed. If the result of a test is abnormal, every part reached by x was an abnormal output. An abnormal output means that either the part <u>or</u> a part that reaches it is abnormal and has failed. This action, in most cases, is the same as reading an instrument or CRT.

2.3.5 MCC Analysis

There are times when MCC provides information unavailable to the shuttle crew. (Mx) is the way to test a specific part of the fuel cell. The result of MCC analysis will be normal or abnormal as in test between parts.

2.3.6 Remaining Parts

Remaining parts (R) - Type an R, or slide the mouse to R, to see a list of parts in the current subsystem that may have failed. The remaining parts list is a list derived by logic based on the tests that have been made previously.

2.3.7 Functional Advice

Functional Advice (F) - Type an F, or slide the mouse to F, to receive advice about the heat logical test to perform. The functional advice is based on previous tests made, which part will provide the most information, etc.

2.3.8 Procedural Advice

Procedural Advice (P) - Type a P, or slide the mouse to P, to receive advice about the next step in the procedure to follow. Procedural advice is based upon the symptoms of the problem and follows, as closely as possible, NASA guidelines.

2.3.9 Answer

Answer (Ax) - When you think that you have identified the part which failed, Ax tells the computer that you believe the answer is part x. If x is the failed part, then you will receive a message stating that you have solved the problem. If x is not the failed part you must continue troubleshooting.

2.3.10 Quit

Quit (Q) - Type a Q, or slide the mouse to Q, to leave the simulation before finding the answer.

2.4 Aiding Messages

At times you may receive messages in the lower display window. There are many different categories of error messages and numerous messages specific to a single component or symptom. Sample messages are:

1. It was unnecessary to observe the fuel cell heat exchanger because the accumulator reaches the fuel cell heat exchanger and the accumulator has a bad output.

2. It was unnecessary that MCC analyze the condenser because you already had MCC analyze the condenser.

3. It was unnecessary to replace the separator pump since you know that it has a good output.

These messages are designed to help you learn to avoid unnecessary diagnostic tests. When you receive such a message, study it to understand why your action was unnecessary. This will help you avoid the same mistakes later.

2.4.1 Completing a Problem

Once you finish a problem by identifying the failed part, or if you quit, you will receive a general performance summary. The performance summary is described in Section 3.0.

3.0 Performance Summary - What Did You Learn?

The feedback that you receive at the end of each problem is very important. It should help you assess your level of understanding of the fuel cell and ability to apply your knowledge to system troubleshooting. The feedback that you receive at the conclusion of each problem will include information on your performance compared to the average score of other simulation users. Figure A-6 shows the feedback that you will receive at the end of each problem.

Congratulations! You have corrected the problem.

Number of students completing this problem:	78
Number of students who quit before solving:	2
Number of students who received a time out:	6

	Your	Average
Number of minutes for diagnosis:	7	8.4
Number of errors made:	2	1.6
Number of displays accessed:	14	17.3
Number of times procedural advice used	1	1.8
Number of times functional advice used	2	1.3

Figure A-6. Performance Summary

3.1 Measures of Performance

When engaged in real equipment troubleshooting in the shuttle, your main goal is to find the problem as quickly as possible. Second, you want to avoid making mistakes that result in longer critical situation times. Therefore, the simulation performance feedback includes information on time to solution and number of mistakes.

3.1.1 Time

The time measures include the actual time that you spent on the simulation. You will also receive information on the average time used by others who have solved the same problem.

3.1.2 Errors

Throughout the simulation, each time you make an error in logic, you immediately receive an error message. The final performance summary is simply a tally of these errors.

3.1.3 Displays Accessed

Your total number of displays accessed is recorded. Though this is not necessarily a good performance measure, it helps you to compare your number of display accesses to those of other simulation users.

3.1.4 Procedural and Functional Advice Used

These are a tally of your use of the two experts. In the beginning, it is likely that your use of advice will exceed the average. After an hour or so, you should be able to solve the problems without using these advice options.

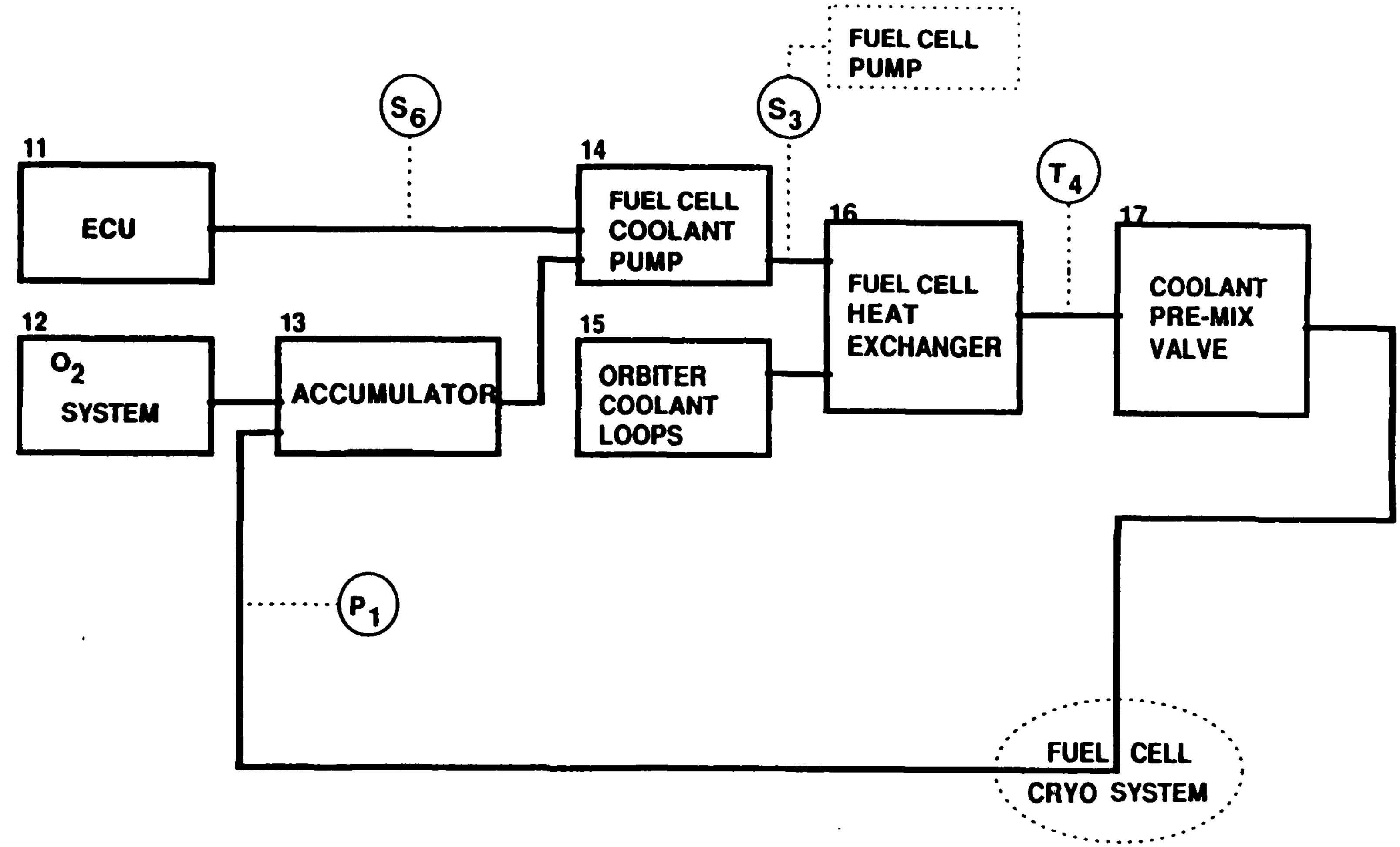
3.2 Summary of Your Learning via the Simulation

After you have used the simulation for a few hours, you will be better prepared for FC troubleshooting in the shuttle. The chances of seeing the same problems on real equipment are slight. However, the information-seeking and problem-solving skills you will develop on the simulation will help you in real world situations.

The simulation will show you the important gauges and controls on the local panel. You can review your system knowledge using the schematics. The simulation will provide you with specific system knowledge and, more importantly, encourage you to take a very logical approach to system troubleshooting. The logic-based diagnostic approach to troubleshooting, as reinforced by this simulation, can be used for problem solving on any technical system.

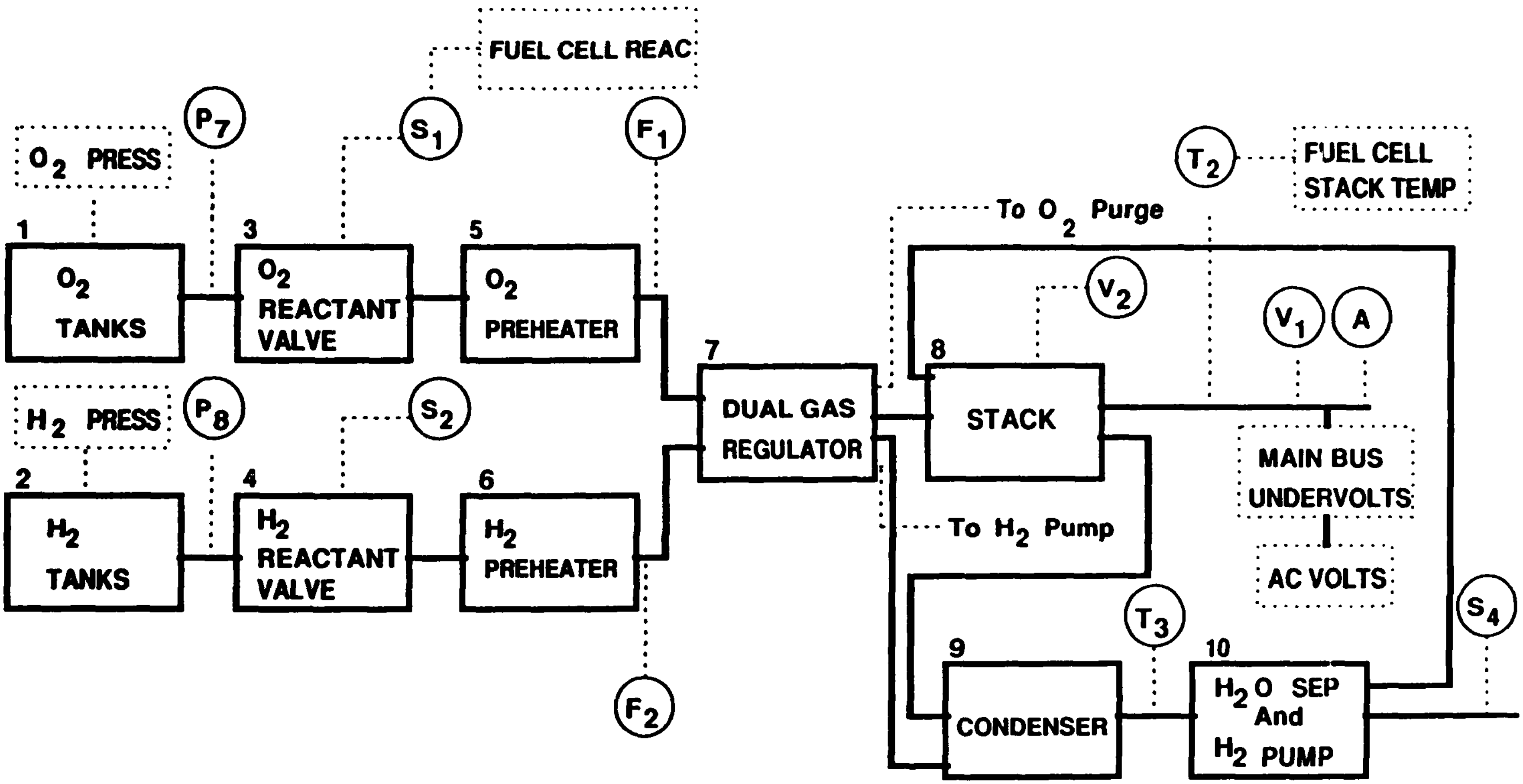
In addition, you learn and practice the procedures established by NASA experts to diagnose problems. This ability to practice solving simulated problems with both functional and procedural feedback is valuable. You gain skills and knowledge that will be invaluable during a real malfunction.

THERMAL CONTROL SYSTEM



GAUGE	NAME	LOCATION(S)	EFFECT
S3	Coolant Pump Tb	R1C, Syssumm1, & Spec 69	T14-16
S6	Rdy for LD Tb	R1C	T11-14
T4	Coolant Temp	Spec 69	T16-17
P1	Coolant Pressure	Spec 69	T13-17

FUEL CELL CRYO SYSTEM



GAUGE	NAME	LOCATION(S)	EFFECT
S1	O2 Reac Vlv Tb	R1C, Syssumm1, & Spec 69	MCC3
S2	H2 Reac Vlv Tb	R1C & Spec 69	MCC4
S4	Ph of H2O	Spec 69	T10-8
F1	O2 Flow Rate	Spec 69	T5-7
F2	H2 Flow Rate	Spec 69	T6-7
V1	Volts	Syssumm1, Spec 67 & Spec 69	T8-9
V2	Volts Monitor/SS2	Syssumm1 & Spec 69	MCC8
T2	Stack Temp	Syssumm1, Spec 69, & O2	T8-9
T3	Exit Temp	Syssumm1 & Spec 69	T9-10
P7	O2 Tank Pressure	Syssumm2, Spec 68 & O2	T1-3
P8	H2 Tank Pressure	Syssumm2, Spec 68 & O2	T2-4
A	Amps	Syssumm1, Spec 67, Spec 69 & F9	T8-9